项目说明文档

数据结构课程设计

——考试报名系统

作 者 姓 名： 郑柯凡

学 号： 1950072

指 导 教 师： 张颖

学院、 专业： 软件学院 软件工程

同济大学

Tongji University

目录

[1 分析 3](#_Toc4629)

[1.1 背景分析 4](#_Toc17529)

[1.2 功能分析 4](#_Toc12657)

[2 设计 4](#_Toc2123)

[2.1 数据结构的选择 4](#_Toc27625)

[2.2 类的设计 4](#_Toc1086)

[3 功能实现 5](#_Toc16034)

[3.1 输入考生信息功能的实现 6](#_Toc24185)

[3.1.1输入考生信息功能流程图 6](#_Toc18013)

[3.1.2输入功能核心代码 6](#_Toc23335)

[3.1.3输入功能错误提示 6](#_Toc3929)

[3.1.4输入功能截屏示例 7](#_Toc21313)

[3.2 查询考生信息功能的实现 7](#_Toc32125)

[3.2.1查询考生信息功能流程图 7](#_Toc5794)

[3.2.2查询功能核心代码 7](#_Toc22171)

[3.2.3查询功能错误提示 8](#_Toc15270)

[3.2.4查询功能截屏示例 8](#_Toc2063)

[3.3 添加考生信息功能的实现 8](#_Toc12523)

[3.3.1添加考生信息功能流程图 8](#_Toc18945)

[3.3.2添加功能核心代码 9](#_Toc18923)

[3.3.3添加功能错误提示 10](#_Toc5656)

[3.3.4添加功能截屏示例 10](#_Toc1411)

[3.4 修改考生信息功能的实现 11](#_Toc21598)

[3.4.1修改考生信息功能流程图 11](#_Toc1449)

[3.4.2修改功能核心代码 11](#_Toc30882)

[3.4.3修改功能错误提示 12](#_Toc10718)

[3.4.4修改功能截屏示例 12](#_Toc23458)

[3.5 删除考生信息功能的实现 13](#_Toc2590)

[3.5.1删除考生信息功能流程图 13](#_Toc14013)

[3.5.2删除功能核心代码 13](#_Toc15963)

[3.5.3删除功能错误提示 14](#_Toc7679)

[3.5.4删除功能截屏示例 14](#_Toc32456)

[3.6 统计(输出)考生信息功能的实现 15](#_Toc29636)

[3.6.1统计考生信息功能流程图 15](#_Toc19322)

[3.6.2统计功能核心代码 15](#_Toc27210)

[3.6.3统计功能截屏示例 15](#_Toc32306)

[3.7 总体系统的实现 16](#_Toc2785)

[3.7.1总体系统功能流程图 16](#_Toc32586)

[3.7.2总体功能核心代码 16](#_Toc14243)

[3.7.3操作码输入错误提示 16](#_Toc24335)

[3.7.4总体功能截屏示例 17](#_Toc979)

[4 测试 17](#_Toc24352)

[4.1 功能测试 17](#_Toc24664)

[4.1.1 输入功能测试 17](#_Toc4997)

[4.1.2 查询功能测试 17](#_Toc17647)

[4.1.3 添加功能测试 18](#_Toc16782)

[4.1.4 修改功能测试 18](#_Toc1031)

[4.1.5 删除功能测试 19](#_Toc4811)

[4.1.6 统计功能测试 19](#_Toc9829)

[4.2 边界测试 20](#_Toc9363)

[4.2.1 输入0个考生信息 20](#_Toc21763)

[4.2.2 删除后有0个考生信息 20](#_Toc246)

**1 分析**

**1.1 背景分析**

由于考生人数的逐年增长，传统的考试报名工作已不能由人手工完成，这给各高校报名工作带来了新的挑战，给教务管理部门增加了很大的工作量。因此，研发一套成熟的功能完善的考试报名系统势在必行。

**1.2 功能分析**

一套成熟完善的考生报名系统需要包含但不仅限于以下功能：输入考生信息；统计（输出）考生信息；查询考生信息；添加考生信息；修改考生信息；删除考生信息。其中考生信息包括准考证号，姓名，性别，年龄和报考类别等信息。

**2 设计**

**2.1 数据结构的选择**

由于考生报名系统需要涉及频繁的增加，删除，修改，查找，若用数组进行存储既不便于空间的开辟，也不便于数据的移动，因此选用增加和删除很方便的链表进行存储，以提高系统运行的效率。同时，为了减少增加和删除操作中的代码量，选用带有表头结点的链表作为存储考生信息的数据结构。

**2.2 类的设计**

此项目中包含三个类，学生对象类，结点类和链表类。

学生对象类的私有成员包括准考证号，姓名，性别，年龄，报考类别。用于存放每个学生的个人信息。公有成员包括构造，析构函数，以及一系列获取私有成员数据的接口函数。

1. //学生对象类
2. **class** Student {
3. **private**:
4. //准考证号，姓名，性别，年龄，报考类别
5. **int** Id;
6. string Name;
7. string Gender;
8. **int** Age;
9. string Type;
10. **public**:
11. Student()
12. {
13. Id = 0; Name = "0"; Gender = "0"; Age = 0; Type = "0";
14. }
15. Student(**int** id, string name, string gender, **int** age, string type)
16. {
17. Id = id; Name = name; Gender = gender;
18. Age = age; Type = type;
19. }
20. ~Student() {};
21. **int** getStuId(){**return** **this**->Id;}
22. string getStuName(){**return** **this**->Name;}
23. string getStuGender(){**return** **this**->Gender;}
24. **int** getStuAge(){**return** **this**->Age;}
25. string getStuType(){**return** **this**->Type;}
26. };

结点类的私有成员包括一个学生对象，用于存放一个学生对象的信息。以及两个指针，一个指向前一个结点的地址，一个指向后一个结点的地址。同时在结点类中也声明了链表类是它的友元类，便于在链表类中直接操作结点的私有成员。

1. //存放考生数据和下个节点地址的节点类
2. **class** Node {
3. **friend** **class** Linklist;
4. **private**:
5. Student rec;
6. Node\* next;
7. Node\* last;
8. Node() { rec = Student(); next = NULL; last = NULL; };
9. ~Node() {};//析构函数，自动释放空间
10. };

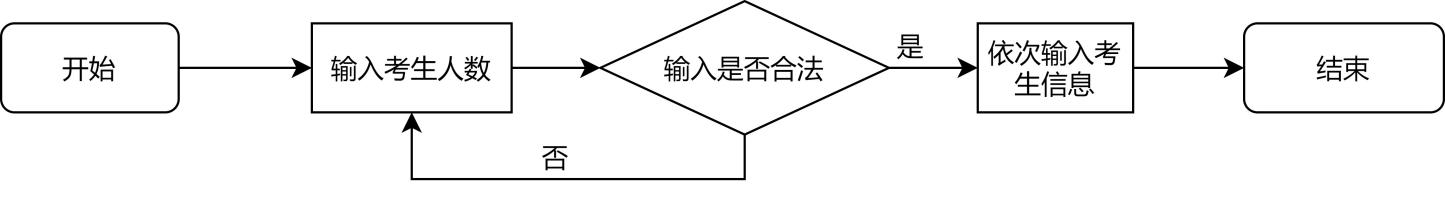
最后是链表类，用于存放数量庞大的考生信息。它的私有成员包括链表长度，用于记录链表的长度，以及指向链表头结点的指针。它的公有成员包括添加，删除，输入，输出等功能函数。

1. //链表类
2. **class** Linklist {
3. **int** size;//链表长度
4. Node\* head;//链表头地址
5. **public**:
6. //构造函数(存在表头结点)
7. Linklist();
8. //析构函数，自动释放空间
9. ~Linklist();
10. //输入考生信息
11. **void** Inport();
12. //输出全部考生信息
13. **void** Display();
14. //查询考生信息
15. **void** Inquire();
16. //添加考生信息
17. **void** Append();
18. //修改考生信息
19. **void** Modify();
20. //删除考生信息
21. **void** Remove();
22. //操作选择函数
23. **void** selectOperation();
24. };

**3 功能实现**

**3.1 输入考生信息功能的实现**

**3.1.1输入考生信息功能流程图**



**3.1.2输入功能核心代码**

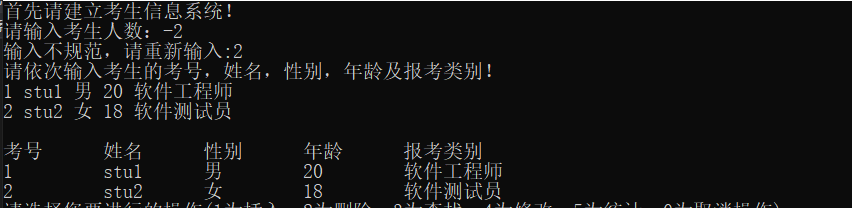
1. //将考生信息输入系统
2. **void** Linklist::Inport()
3. {
4. cout << "首先请建立考生信息系统！" << endl;
5. **int** studentnum = 0;
6. cout << "请输入考生人数：";
7. **while** (1)
8. {
9. cin >> studentnum;
10. **if** (!cin.good()||studentnum<0)
11. {
12. cout << "输入不规范，请重新输入:" ;
13. cin.clear();
14. cin.ignore(1024, '\n');
15. }
16. **else**
17. {
18. **break**;
19. }
20. }
21. cin.get();
22. **this**->size = studentnum;
23. **if** (studentnum == 0) { **return**; }
24. cout << "请依次输入考生的考号，姓名，性别，年龄及报考类别！" << endl;
25. **int** id = 0; string name; string gender; **int** age = 0; string type;
26. Node\* current = **this**->head;
27. Node\* tempnode;
28. **for** (**int** i = 0; i < studentnum; i++)
29. {
30. //开辟空间
31. tempnode = current;
32. current->next = **new** Node;
33. current = current->next;
34. current->last = tempnode;
35. //输入数据
36. cin >> id >> name >> gender >> age >> type;
37. current->rec = Student(id, name, gender, age, type);
38. }
39. cout << endl;
40. }

**3.1.3输入功能错误提示**

当输入不合法的考生人数时（考生人数为负数），将给出错误提示：输入不规范，并要求重新输入正确的考生人数，程序能正常运行不崩溃。

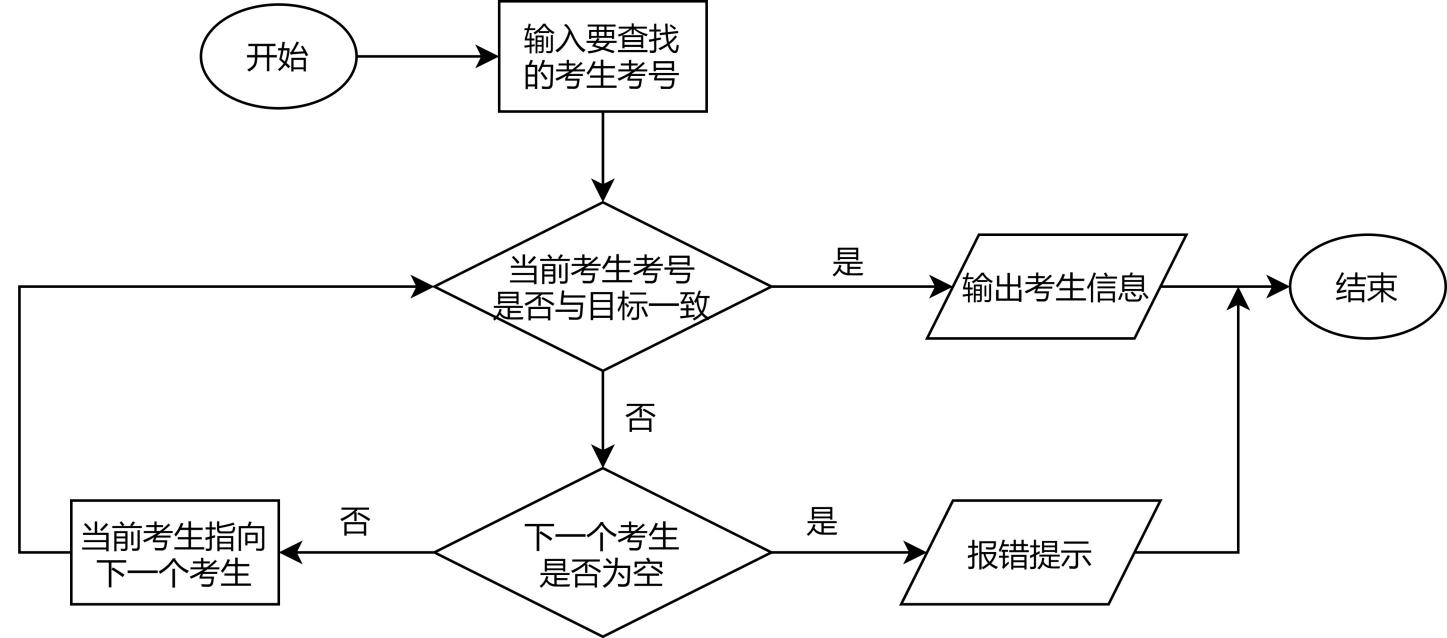
(具体运行情况见下条截屏示例)

**3.1.4输入功能截屏示例**



**3.2 查询考生信息功能的实现**

**3.2.1查询考生信息功能流程图**



**3.2.2查询功能核心代码**

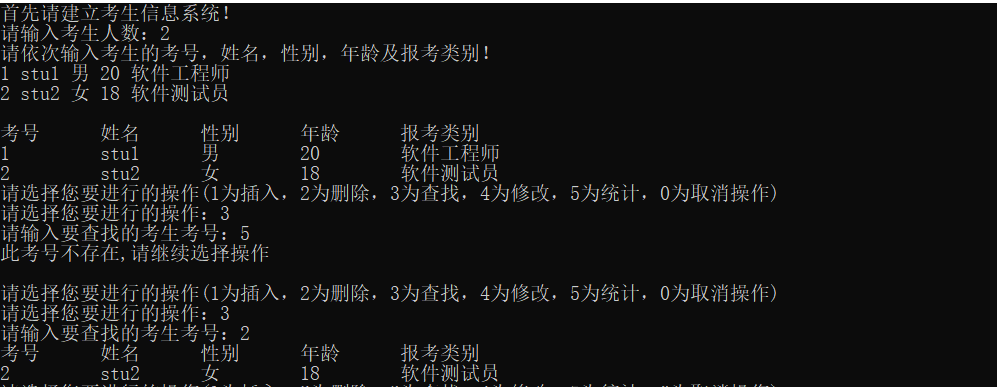
1. //查询考生信息
2. **void** Linklist::Inquire()
3. {
4. **int** num = 0;
5. cout << "请输入要查找的考生考号：";
6. cin >> num;
7. Node\* temp = **this**->head->next;
8. **while** (temp != NULL)
9. {
10. **if** (temp->rec.getStuId() == num)
11. {
12. cout << setiosflags(ios::left);
13. cout << setw(10) << "考号" << setw(10) << "姓名" << setw(10) << "性别" << setw(10) << "年龄" << setw(10) << "报考类别" << endl;
14. cout << setw(10) << temp->rec.getStuId() << setw(10) << temp->rec.getStuName() << setw(10) << temp->rec.getStuGender() << setw(10) << temp->rec.getStuAge() << setw(10) << temp->rec.getStuType() << endl;
15. cout << endl;
16. **return**;
17. }
18. **else** **if** (temp->rec.getStuId() != num)
19. {
20. temp = temp->next;
21. }
22. }
23. cout << "此考号不存在,请继续选择操作" << endl;
24. cout << endl;
25. **return**;
26. }

**3.2.3查询功能错误提示**

当输入不合法的考生考号时（即系统中不存在的考生考号），将给出错误提示：考号不存在，并要求使用者继续选择其他操作，程序能正常运行不崩溃。

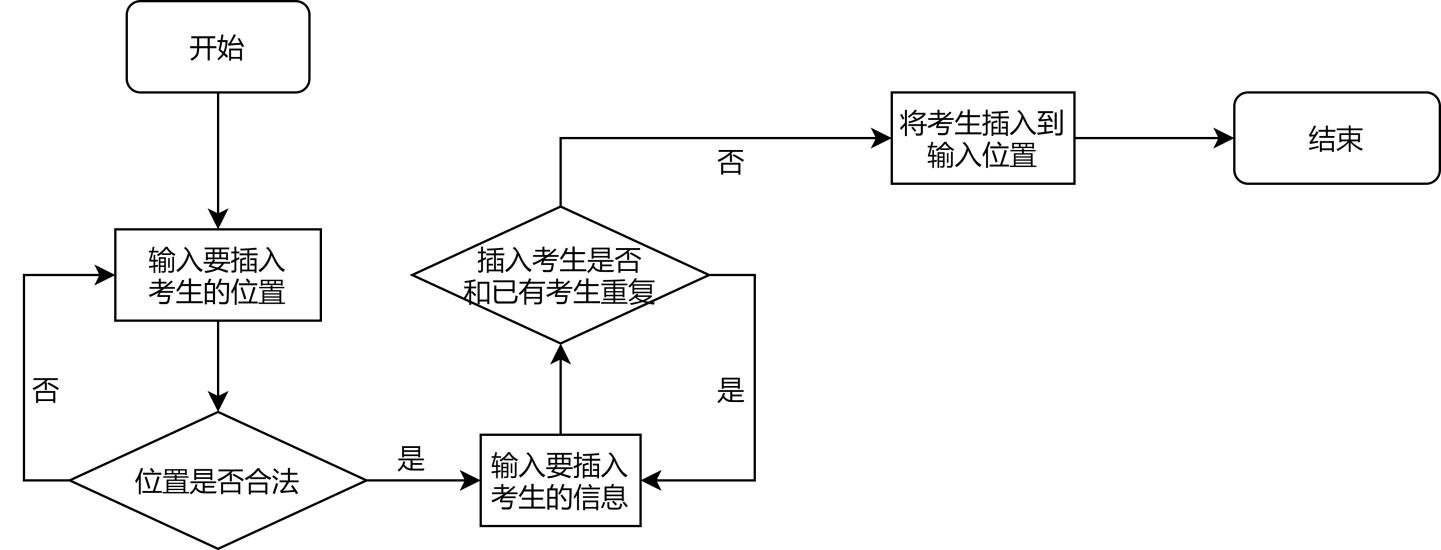
(具体运行情况见下条截屏示例)

**3.2.4查询功能截屏示例**



**3.3 添加考生信息功能的实现**

**3.3.1添加考生信息功能流程图**



**3.3.2添加功能核心代码**

1. //添加考生信息
2. **void** Linklist::Append()
3. {
4. **int** num = 0;
5. cout << "请输入要插入的考生位置：";
6. cin >> num; cin.get();
7. **if** (num <= 0 || num > **this**->size + 1)
8. {
9. cout << "输入不规范，请重新选择操作" << endl;
10. cout << endl;
11. **return**;
12. }
13. cout << "请依次输入考生的考号，姓名，性别，年龄及报考类别！" << endl;
14. **int** id = 0; string name; string gender; **int** age = 0; string type;
15. cin >> id >> name >> gender >> age >> type;
16. //检测是否有重复
17. Node\* temp = **this**->head->next;
18. **while** (temp != NULL)
19. {
20. **if** (temp->rec.getStuId() == id)
21. {
22. cout << "考生考号重复，请重新选择操作" << endl;
23. cout << endl;
24. **return**;
25. }
26. **else** **if** (temp->rec.getStuId() != id)
27. {
28. temp = temp->next;
29. }
30. }
31. //插入
32. temp = **this**->head;
33. **while** (num != 1)
34. {
35. temp = temp->next;
36. num--;
37. }
38. **if** (temp->next == NULL)
39. {
40. temp->next = **new** Node;
41. temp->next->rec = Student(id, name, gender, age, type);
42. temp->next->last = temp;
43. temp->next->next = NULL;
44. **this**->size++;
45. cout << "操作成功！" << endl;
46. cout << endl;
47. }
48. **else** **if** (temp->next != NULL)
49. {
50. Node\* extra = **new** Node;
51. extra->rec = Student(id, name, gender, age, type);
52. extra->next = temp->next;
53. extra->last = temp;
54. temp->next = extra;
55. extra->next->last = extra;
56. **this**->size++;
57. cout << "操作成功！" << endl;
58. cout << endl;
59. }
60. }

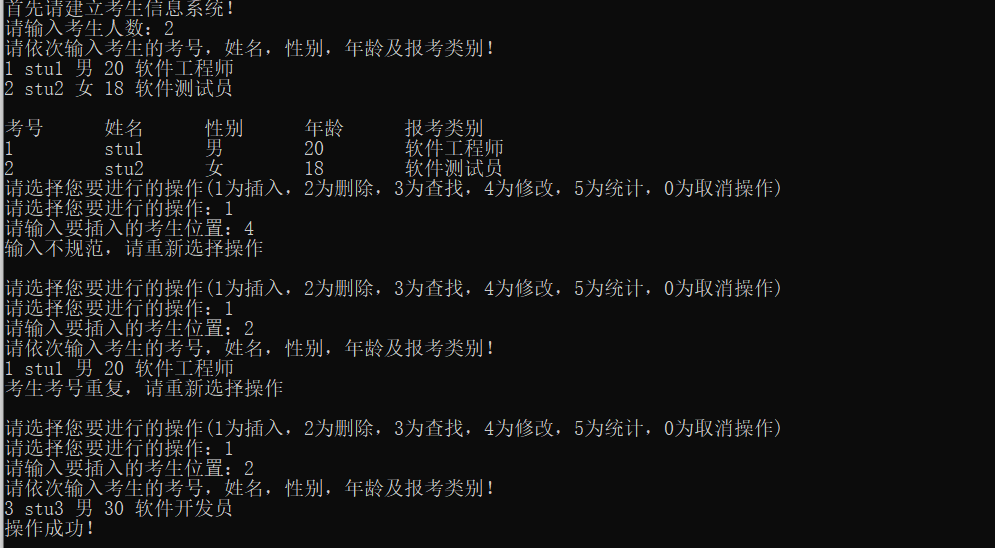
**3.3.3添加功能错误提示**

当输入不合法的插入位置时（即插入位置为负数，或超出系统内人数数量+1），将给出错误提示：输入位置不存在，并要求使用者继续选择其他操作，程序能正常运行不崩溃。

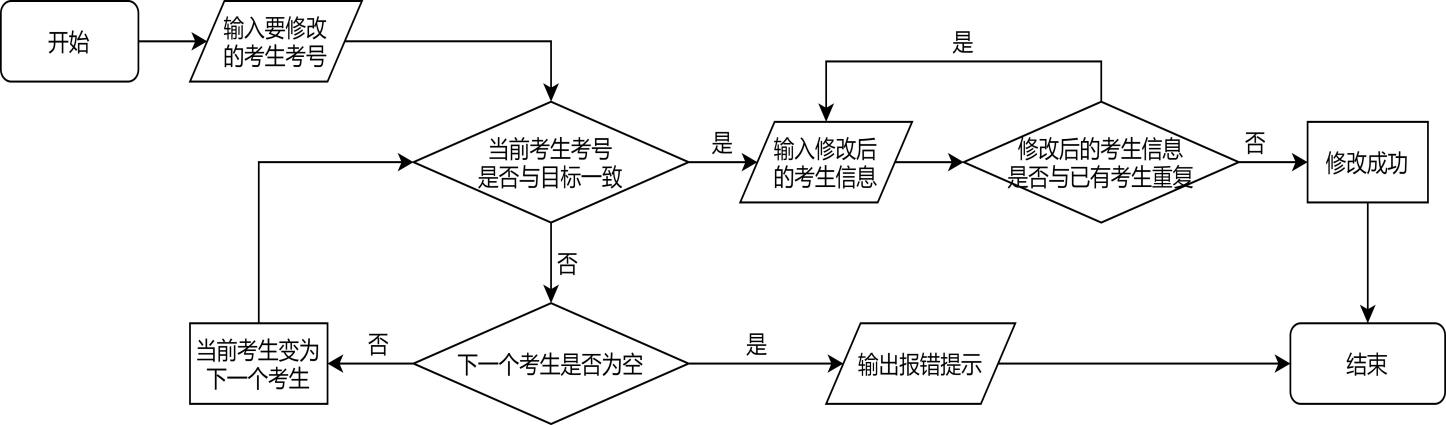
当添加不合法的考生考号时（即要添加的考生考号与系统内其他考生考号存在重复情况），将给出错误提示：考生考号重复，并要求使用者继续选择其他操作，程序能正常运行不崩溃。

(具体运行情况见下条截屏示例)

**3.3.4添加功能截屏示例**



**3.4 修改考生信息功能的实现**

**3.4.1修改考生信息功能流程图**

**3.4.2修改功能核心代码**

1. //修改考生信息
2. **void** Linklist::Modify()
3. {
4. **int** num = 0;
5. cout << "请输入要修改的考生考号：";
6. cin >> num;
7. Node\* temp = **this**->head->next;
8. **while** (temp != NULL)
9. {
10. **if** (temp->rec.getStuId() == num)
11. {
12. cout << "请依次输入修改后考生的考号，姓名，性别，年龄及报考类别！" << endl;
13. **int** id = 0; string name; string gender; **int** age = 0; string type;
14. cin >> id >> name >> gender >> age >> type;
15. //检测是否有重复
16. Node\* temppp = **this**->head->next;
17. **while** (temppp != NULL)
18. {
19. **if** (temppp->rec.getStuId() == id&&temppp!=temp)
20. {
21. cout << "考生考号重复，请重新选择操作" << endl;
22. cout << endl;
23. **return**;
24. }
25. **else**
26. {
27. temppp = temppp->next;
28. }
29. }
30. //进行修改
31. temp->rec = Student(id, name, gender, age, type);
32. cout << "操作成功！" << endl;
33. cout << endl;
34. **return**;
35. }
36. **else** **if** (temp->rec.getStuId() != num)
37. {
38. temp = temp->next;
39. }
40. }
41. cout << "此考号不存在,请继续选择操作" << endl;
42. cout << endl;
43. **return**;
44. }

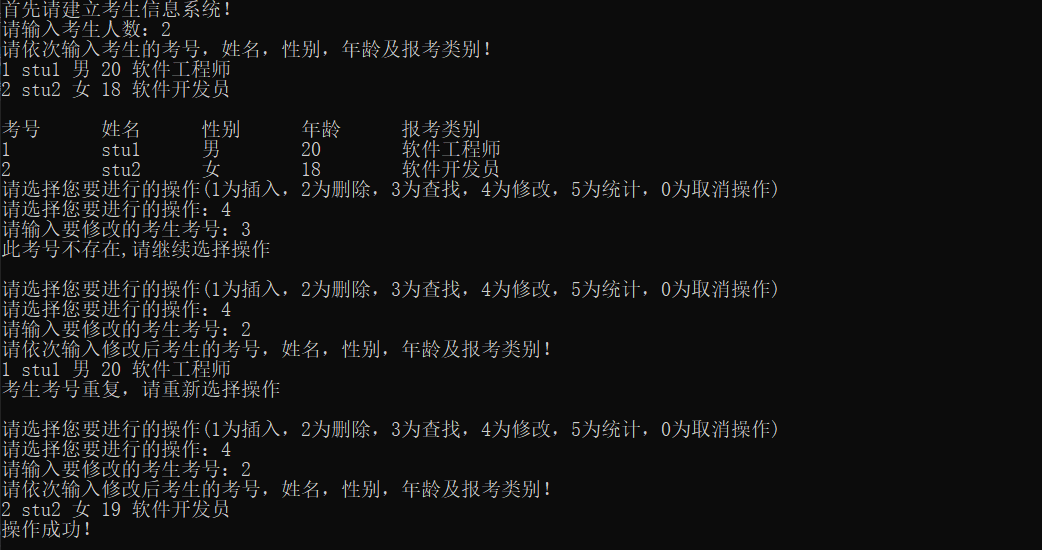
**3.4.3修改功能错误提示**

当要修改的考生考号不合法时（即系统中不存在该考生考号），系统将给出错误提示：考号不存在，并要求使用者继续选择其他操作，程序能正常运行不崩溃。

当修改后的考生考号不合法时（即修改后的考生考号与其他已有考生重复时），系统将给出错误提示：考生考号重复，并要求使用者继续选择其他操作，程序能正常运行不崩溃。

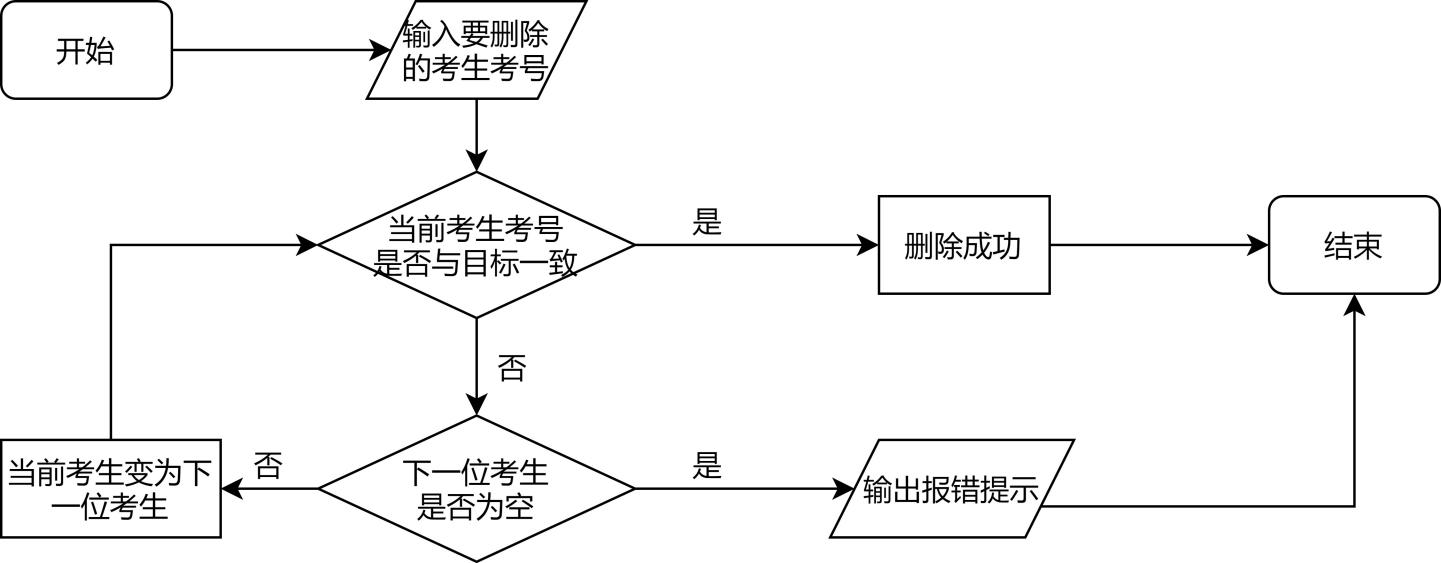
(具体运行情况见下条截屏示例)

**3.4.4修改功能截屏示例**



**3.5 删除考生信息功能的实现**

**3.5.1删除考生信息功能流程图**



**3.5.2删除功能核心代码**

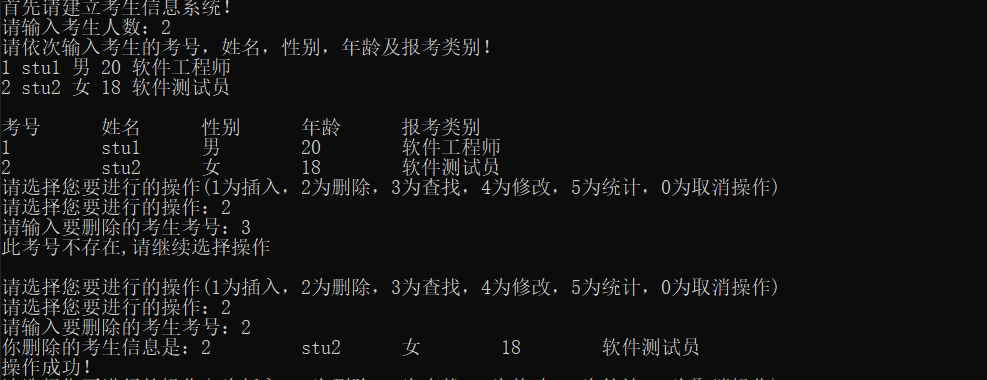
1. //删除考生信息
2. **void** Linklist::Remove()
3. {
4. **int** num = 0;
5. cout << "请输入要删除的考生考号：";
6. cin >> num;
7. Node\* temp = **this**->head->next;
8. **while** (temp != NULL)
9. {
10. **if** (temp->rec.getStuId() == num)
11. {
12. cout << "你删除的考生信息是：";
13. cout << setiosflags(ios::left);
14. cout << setw(10) << temp->rec.getStuId() << setw(10) << temp->rec.getStuName() << setw(10) << temp->rec.getStuGender() << setw(10) << temp->rec.getStuAge() << setw(10) << temp->rec.getStuType() << endl;
15. **if** (temp->next == NULL)
16. {
17. temp->last->next = NULL;
18. **delete** temp;
19. }
20. **else** **if** (temp->next != NULL)
21. {
22. temp->next->last = temp->last;
23. temp->last->next = temp->next;
24. **delete** temp;
25. }
26. cout << "操作成功！" << endl;
27. **this**->size--;
28. **return**;
29. }
30. **else** **if** (temp->rec.getStuId() != num)
31. {
32. temp = temp->next;
33. }
34. }
35. cout << "此考号不存在,请继续选择操作" << endl;
36. cout << endl;
37. **return**;
38. }

**3.5.3删除功能错误提示**

当要删除的考生考号不合法时（即系统中不存在该考生考号），系统将给出错误提示：考号不存在，并要求使用者继续选择其他操作，程序能正常运行不崩溃。

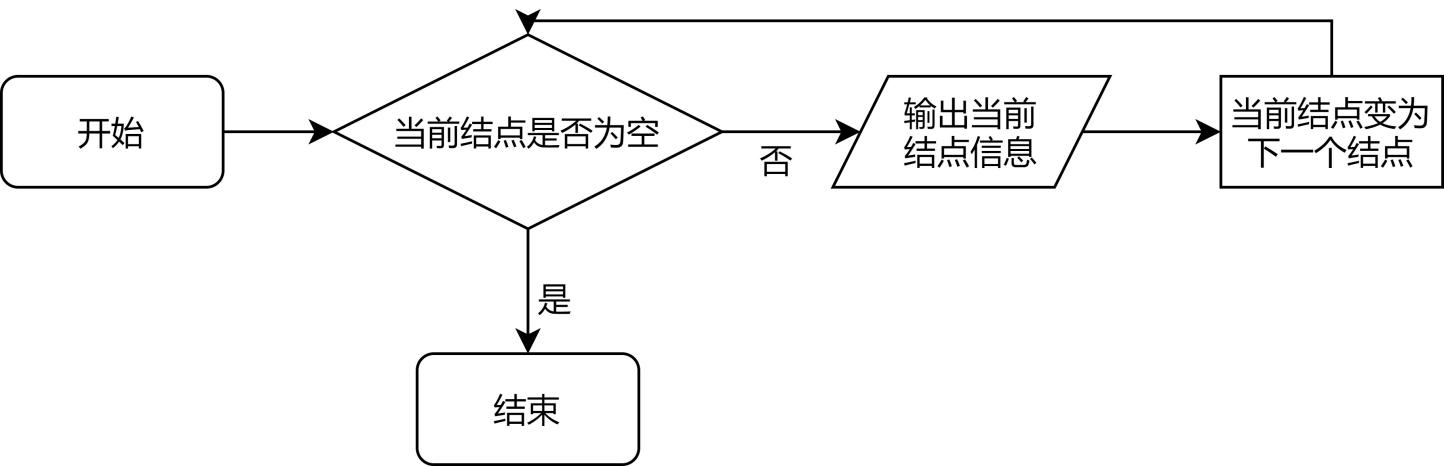
(具体运行情况见下条截屏示例)

**3.5.4删除功能截屏示例**



**3.6 统计(输出)考生信息功能的实现**

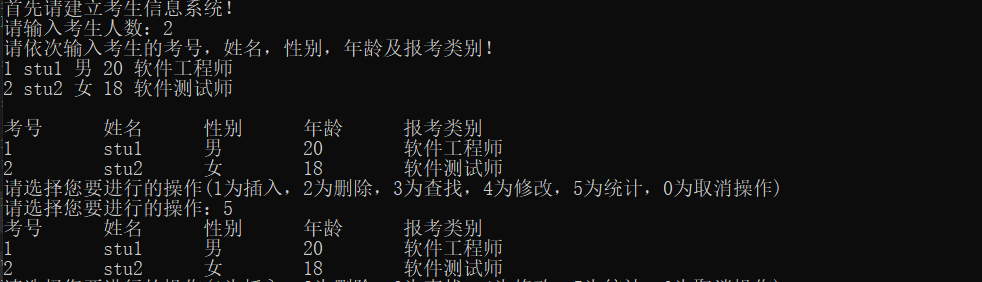
**3.6.1统计考生信息功能流程图**



**3.6.2统计功能核心代码**

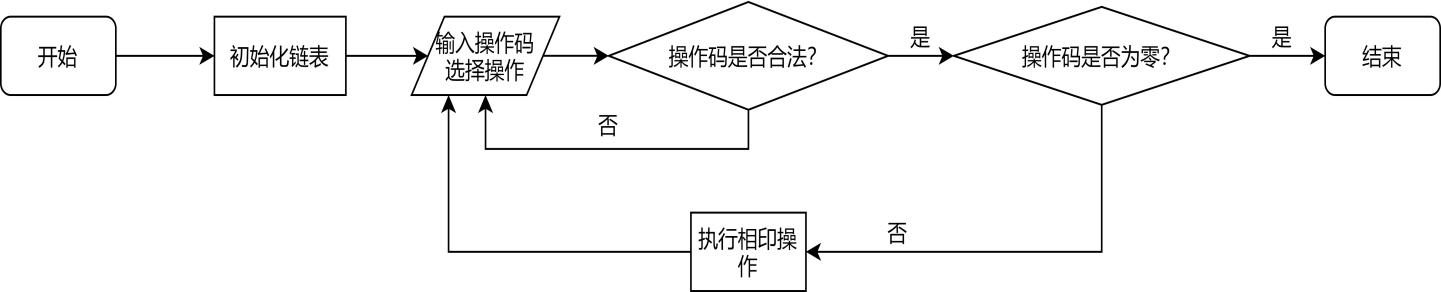
1. //统计考生名单
2. **void** Linklist::Display()
3. {
4. Node\* temp = **this**->head->next;
5. **if** (temp == NULL)
6. {
7. cout << "无考生信息" << endl;
8. }
9. **else**
10. {
11. cout << setiosflags(ios::left);
12. cout << setw(10) << "考号" << setw(10) << "姓名" << setw(10) << "性别" << setw(10) << "年龄" << setw(10) << "报考类别" << endl;
13. **while** (temp != NULL)
14. {
15. cout << setw(10) << temp->rec.getStuId() << setw(10) << temp->rec.getStuName() << setw(10) << temp->rec.getStuGender() << setw(10) << temp->rec.getStuAge() << setw(10) << temp->rec.getStuType() << endl;
16. temp = temp->next;
17. }
18. }
19. cout << endl;
20. }

**3.6.3统计功能截屏示例**



**3.7 总体系统的实现**

**3.7.1总体系统功能流程图**



**3.7.2总体功能核心代码**

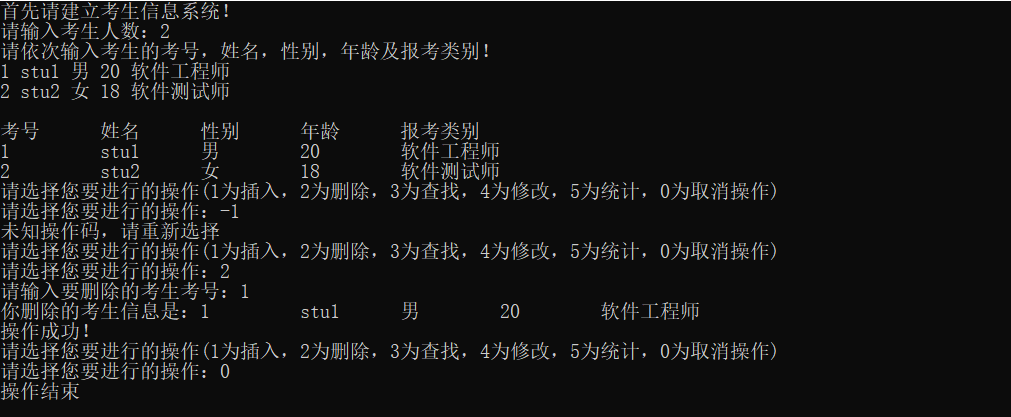
1. **void** Linklist::selectOperation()
2. {
3. **int** operation = 5;
4. **while** (operation!=0)
5. {
6. **switch** (operation)
7. {
8. **case** 1:**this**->Append(); **break**;
9. **case** 2:**this**->Remove(); **break**;
10. **case** 3:**this**->Inquire(); **break**;
11. **case** 4:**this**->Modify(); **break**;
12. **case** 5:**this**->Display(); **break**;
13. **default**:cout << "未知操作码，请重新选择" << endl; **break**;
14. }
15. cout << "请选择您要进行的操作(1为插入，2为删除，3为查找，4为修改，5为统计，0为取消操作)" << endl;
16. cout << "请选择您要进行的操作：";
17. cin >> operation;
18. }
19. cout << "操作结束" << endl;
20. }

**3.7.3操作码输入错误提示**

当输入的操作码不合法时（即不存在对应功能的操作码时），系统将给出错误提示：未知操作码，并要求使用者继续选择其他操作，程序能正常运行不崩溃。

(具体运行情况见下条截屏示例)

**3.7.4总体功能截屏示例**



**4 测试**

**4.1 功能测试**

**4.1.1 输入功能测试**

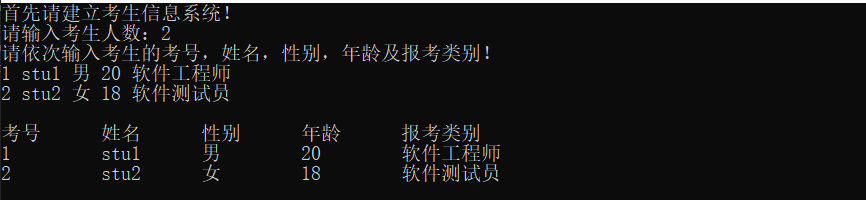
**测试用例：**1 stu1 男 20 软件工程师

2 stu2 女 18 软件测试员

**预期结果：**1 stu1 男 20 软件工程师

2 stu2 女 18 软件测试员

**实验结果：**



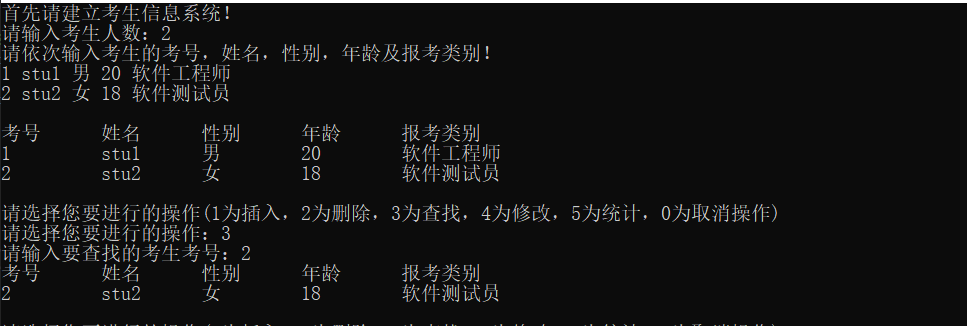
**4.1.2 查询功能测试**

**测试用例：**1 stu1 男 20 软件工程师

2 stu2 女 18 软件测试员

**预期结果：**2 stu2 女 18 软件测试员

**实验结果：**



**4.1.3 添加功能测试**

**测试用例：**1 stu1 男 20 软件工程师

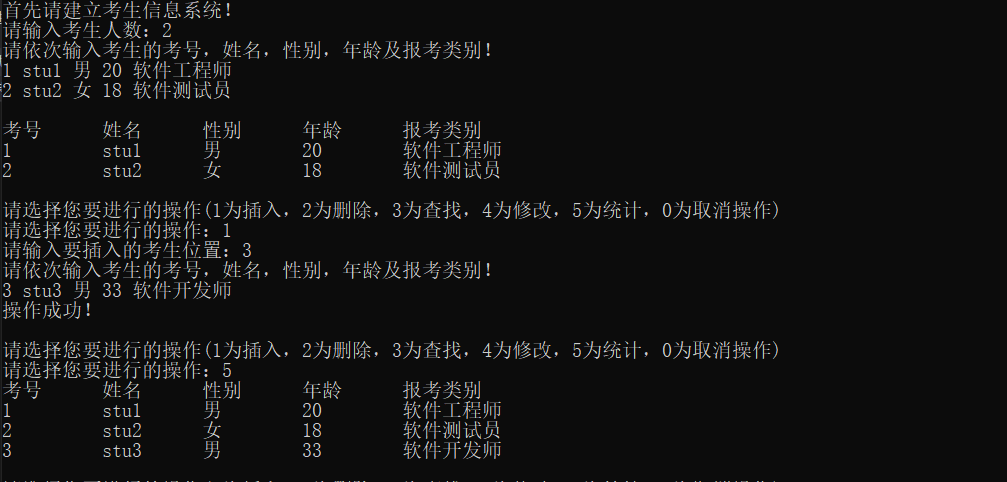
2 stu2 女 18 软件测试员

**预期结果：**1 stu1 男 20 软件工程师

2 stu2 女 18 软件测试员

3 stu3 男 33 软件开发师

**实验结果：**



**4.1.4 修改功能测试**

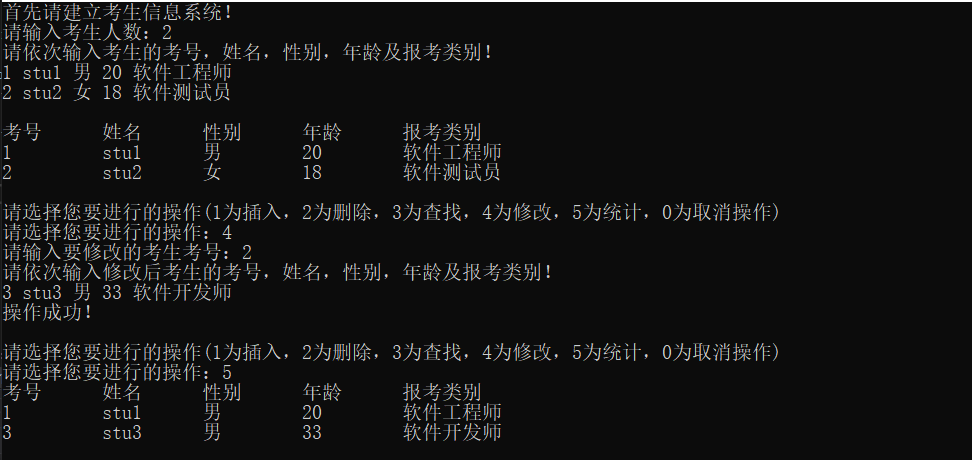
**测试用例：**1 stu1 男 20 软件工程师

2 stu2 女 18 软件测试员

**预期结果：**1 stu1 男 20 软件工程师

3 stu3 男 33 软件开发师

**实验结果：**



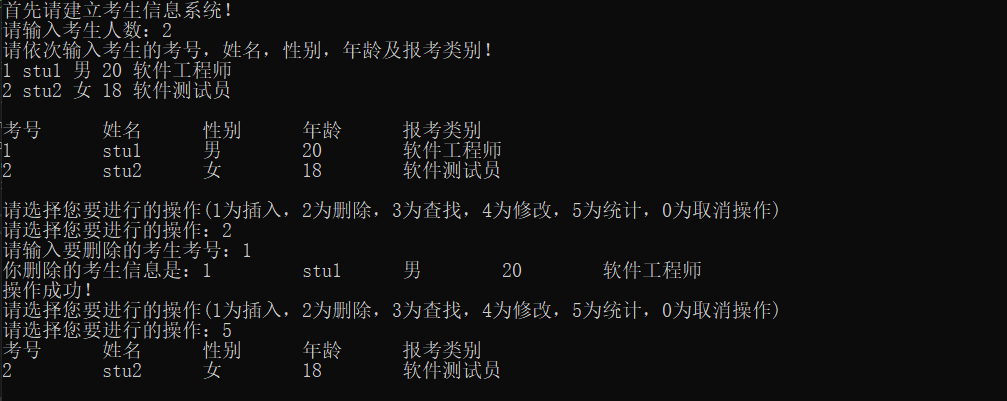
**4.1.5 删除功能测试**

**测试用例：**1 stu1 男 20 软件工程师

2 stu2 女 18 软件测试员

**预期结果：**2 stu2 女 18 软件测试员

**实验结果：**



**4.1.6 统计功能测试**

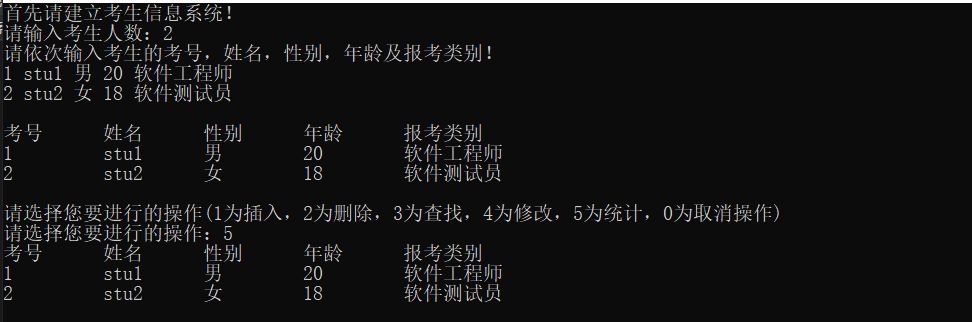
**测试用例：**1 stu1 男 20 软件工程师

2 stu2 女 18 软件测试员

**预期结果：**1 stu1 男 20 软件工程师

2 stu2 女 18 软件测试员

**实验结果：**



**4.2 边界测试**

**4.2.1 输入0个考生信息**

**测试用例：**（无）

**预期结果：**无考生信息

**实验结果：**



**4.2.2 删除后有0个考生信息**

**测试用例：**1 stu1 男 20 软件工程师

**预期结果：**无考生信息

**实验结果：**

